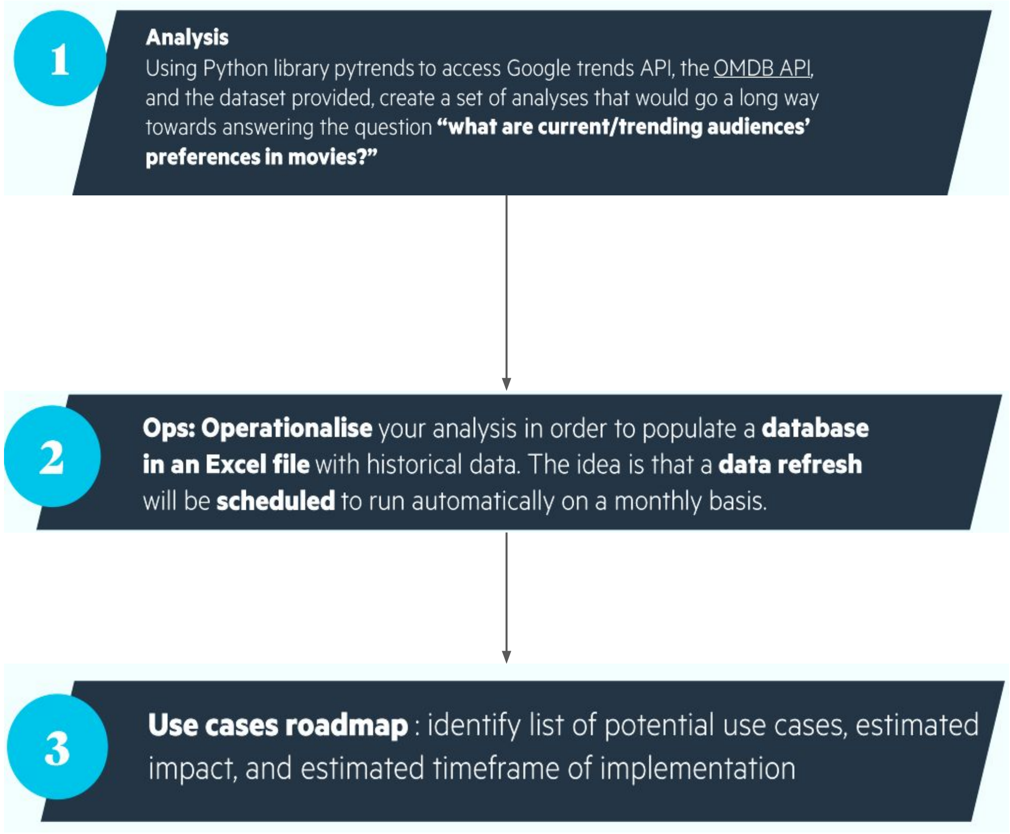
Contents:


1
Analysis
Using Python library pytrends to access Google trends API, the OMDB API, and the dataset provided, create a set of analyses that would go a long way towards answering the question "what are current/trending audiences' preferences in movies?"
2
Ops: Operationalise your analysis in order to populate a database in an Excel file with historical data. The idea is that a data refresh will be scheduled to run automatically on a monthly basis.
3
Use cases roadmap : identify list of potential use cases, estimated impact, and estimated timeframe of implementation

Analysis

1.1 analysis.py

OUTLINE:

- analysis.py, automates the process of fetching, cleaning, and enhancing movie data by using OMDB API and pytrends for Google Trends.
- Reads a list of movie titles from an Excel file, fetches their details from the OMDB API, and processes the data (filtering out movies released before 2007, without a years worth of google interest data or with missing information).
- For each valid movie, the script calculates the average Google search interest over the year following its release date using the pytrends library.
- Saves the cleaned dataset as a CSV file while logging progress and errors throughout the execution.

Analysis

1.2 Pytrends Error 429

- Having issues with using pytrends.
- Google was likely blocking IP as it thought I was a bot, seen similar issues from users online.
- Seen solution online that simulated a legitimate browser session when making requests to Google Trends via package called Selenium.
- Retrieve a specific cookie to enable authenticated requests, reduces the risk of being blocked or receiving incomplete data from Google Trends.

Function Used:
(found online)

```python
def get_cookie() -> str:
    """
    Retrieves the 'NID' cookie value from Google Trends.
    """
    options = webdriver.ChromeOptions()
    options.add_argument("--headless")
    driver = webdriver.Chrome(options=options)
    driver.get("https://trends.google.com/")
    time.sleep(5)
    cookie = driver.get_cookie("NID")
    driver.quit()
    if cookie:
        return cookie["value"]
    logging.error("Failed to retrieve NID cookie")
    return ""
```

Analysis

1.3 Anchoring (pytrends)

- Pytrends gives term search interest relative to its own maximum.

  For example: *If you run a query individually for Movie A and Movie B and they both have interest 100. You can't know whether Movie A actually had more overall searches than Movie B, because each is scaled to its own maximum in individual query.*

- Pytrends allows comparison of interest between terms, but only between 5 terms at a time.
- Therefore to compare the google search interest between a large database of films we must anchor each film's relative interest to an anchor word.
- I chose term 'Feature film' as the anchor word, essential for a word that has constant google trend interest.
- Gives me google interest data which is relative ratio of volume of google searches between the movie and 'Feature film' in the specified timeframe.

Analysis

1.4 More Details

- Standardised google interest search terms by appending 'movie' to all titles (eg Macbeth -> Macbeth Movie) to make google interest data consistent.
- Algorithm makes 5 attempts to retrieve pytrends data, if not it is skipped.
- Saves the cleaned dataset as 'movies_analysis.csv' for further use.

Sample Output:

```
2025-02-23 17:57:15,568 - INFO - Skipping movie 'Junebug' due to incomplete or invalid data.
2025-02-23 17:57:15,569 - INFO - Processing movie: The Phantom
2025-02-23 17:57:15,615 - INFO - Skipping movie 'The Phantom' due to incomplete or invalid data.
2025-02-23 17:57:15,615 - INFO - Processing movie: A Hidden Life
2025-02-23 17:57:16,146 - INFO - Appended data for 'A Hidden Life Movie'.
2025-02-23 17:57:16,146 - INFO - Processing movie: National Treasure: Book of Secrets
2025-02-23 17:57:16,714 - INFO - Appended data for 'National Treasure: Book of Secrets Movie'.
2025-02-23 17:57:16,714 - INFO - Processing movie: I Am Mother
2025-02-23 17:57:16,780 - INFO - Skipping movie 'I Am Mother' due to incomplete or invalid data.
2025-02-23 17:57:16,780 - INFO - Processing movie: Awake
2025-02-23 17:57:17,351 - INFO - Appended data for 'Awake Movie'.
2025-02-23 17:57:17,352 - INFO - Processing movie: Book of Dragons
2025-02-23 17:57:17,406 - INFO - Skipping movie 'Book of Dragons' due to incomplete or invalid data.
2025-02-23 17:57:17,407 - INFO - Processing movie: What the Health
2025-02-23 17:57:17,461 - INFO - Skipping movie 'What the Health' due to incomplete or invalid data.
2025-02-23 17:57:17,483 - INFO - CSV saved to movies_updated.csv.
```

Analysis

1.5 Output

- movie_anaylsis.csv created once data cleaned and enriched.
- Box office revenue data tend to be skewed because only a few films, often blockbusters, earn very high revenues, while the majority of films generate relatively modest earnings. This creates a long right tail in the distribution. Therefore due to skew, apply log transformation when modelling (& before correlations).

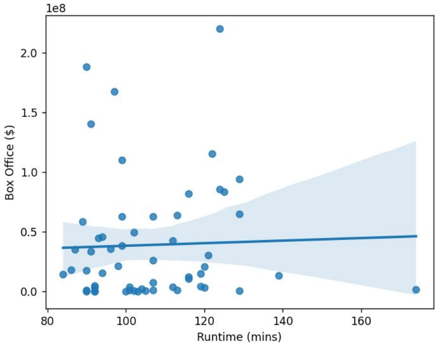
Sample CSV:

Title	Year	Runtime (n	IMDb Ratin	IMDb Vote:	Box Office	Age Rating	Google Interest
The Age of	2015	112	7.2	212877	42629776	PG-13	1.139410188
Friends wit	2011	107	6.1	43369	7251073	R	0.352269312
Girls Trip	2017	122	6.2	41464	1.15E+08	R	2.44047619
The Art of t	2013	90	6.3	26330	64065	R	0.085118067
Macbeth	2015	113	6.6	60210	1110707	R	0.828643216
London Ha	2016	99	5.9	174344	62524260	R	2.625
Anthropoic	2016	120	7.2	55712	2964845	R	0.336814621
A Quiet Pla	2018	90	7.5	620408	1.88E+08	PG-13	2.057208238
God's Not I	2016	120	4.3	13832	20774575	PG	0.541549296
Hamlet 2	2008	92	6.3	17453	4886216	R	0.109657158
Ashby	2015	100	6.4	16349	4631	R	0.350776164
Middle Mei	2009	105	6.8	38590	754301	R	0.350376749
Ginger & R	2012	90	6.2	11877	1012973	PG-13	0.000957854
Hotel Tran:	2018	97	6.3	90465	1.68E+08	PG	0.056958128
Song to Soi	2017	129	5.6	23002	443684	R	7.840080972
The Art of S	2019	104	6.6	41578	2410914	R	0.040767386
Ninja Assa	2009	99	6.3	76573	38122883	R	1.23723229
Mechanic:	2016	98	5.7	96852	21218403	R	1.104247104
The Huntin	2007	101	6.8	26288	969869	R	0.128582494
Certain Wc	2016	107	6.4	16105	1087585	R	0.047658176
The Post	2017	116	7.2	165772	81903458	PG-13	2.320512821
Now You S	2016	129	6.4	333462	65075540	PG-13	4.132352941
Bedtime St	2008	99	6	102992	1.1E+08	PG	1.150313152
Saving Mr.	2013	125	7.5	172080	83301580	PG-13	0.151218063
Elegy	2008	112	6.7	23406	3581642	R	0.485217391
Welcome t	2018	116	6.2	26683	10763520	PG-13	0.332982086
Berberian :	2012	92	6.2	17777	38493	Not Rated	0.001253133
Frankenwe	2012	87	6.9	120955	35291068	PG	0.495847176
Unknown	2011	113	6.8	271966	63686397	PG-13	3.745874587

Analysis

1.6 Correlation Insights

1. Do longer films gross more?




```
> cor.test(runtime, log(box_office))

        Pearson's product-moment correla

data:  runtime and log(box_office)
t = 0.61783, df = 54, p-value = 0.5393
alternative hypothesis: true correlation
95 percent confidence interval:
 -0.1831542  0.3392096
sample estimates:
       cor
0.08378048
```

2. Correlation between IMDb votes and ratings?

```
> cor.test(imdb_votes, imdb_rating)

        Pearson's product-moment correlation

data:  imdb_votes and imdb_rating
t = 3.2788, df = 54, p-value = 0.001828
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.1619111 0.6055041
sample estimates:
      cor
0.4074672
```

3. Correlation between google searches and IMDb rating?

```
> cor.test(google_interest, imdb_rating)

        Pearson's product-moment correlation

data:  google_interest and imdb_rating
t = -0.69381, df = 54, p-value = 0.4908
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.3482905  0.1731833
sample estimates:
       cor
-0.0939969
```

4. Correlation between google searches and box office earnings?

```
> cor.test(google_interest, log(box_office))

        Pearson's product-moment correlation

data:  google_interest and log(box_office)
t = 1.8031, df = 54, p-value = 0.07695
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.02624338  0.47165298
sample estimates:
      cor
0.2383013
```

Operationalise

2.1 Outline

- Create master Excel file, and schedule monthly to append new movies released over a year ago (in order to have years worth of google interest data).
- Again fetches OMDb and pytrend data.
- Ensures only new, valid movies are added to build a cumulative, historical dataset.
- By running monthly, prepares an enriched dataset for advanced analyses like large-scale Marketing Mix Modeling (MMM).

Operationalise

2.2 movie_gather.py

- In order to gather movie data for all movies released after 2006 and that have a years worth of google interest data I used the IMDb Non-Commercial Dataset (link).
- This allows me to download title.basics.tsv.gz which is a log of all titles and release years of movies on IMDb. It however does not have release date month (or other details so OMDb API still necessary).
- Using this I made movie_gather.py which uses pandas to filter through all movies in the database, selecting movies released between 2007 and last year.
- It then adds all these movies to movies_gather.xlsx.
- It appended 295,822 movies.

- *perhaps after initial gathering of historical data, when this is scheduled to run monthly new database needs to be downloaded, and could be changed to only look for movies released last year, so less memory intensive on operationalise.py.

Operationalise

2.3 operationalise.py

- Once gathering all movie data, similar to analysis.py, operationalise.py gathers all OMDb data and pytrend data for movies.
- Here it checks if there is years worth of google trend data, as previous database only had release year not month.
- Cleans data similar to analysis.py
- Appends cleaned data to movies_master.xlsx to produce growing database of movies.
- Due to pytrend unreliability I did not append all ~ 300,000 movies. However potential is there.

Use Cases

3.1 Neon Films' Roadmap

Stage 1: Data Foundation & Basic Insights

Movie Data Pipeline

Process & clean movie data (OMDb API and Google Trends)

Build a movie database for historical trend analysis that updates every month.

Insights & Correlation

Run initial correlations (eg runtime vs box office, search interest vs IMDb rating)

Create dashboards/reports for quick-win insights

Timeframe: Immediate

Impact: Low– Likely just confirms intuitions in movie trends.

Stage 2: Predictive Modeling & Marketing Optimisation

Marketing Mix Modeling (MMM)

Use movie database to create MMM.

Consider Neon Films' aims and capabilities also integrate additional data about Neon Films such as company spending (eg in marketing) to make more robust models.

Timeframe: Short (<1 Month)

Impact: Medium– Informs strategic marketing and release decisions

Stage 3: Strategic Decision-Making & Personalisation

Trend-Based Content Greenlighting

Analyse emerging genres / themes in current google trend data.

Match internal dashboards, in order to decide what content to create.

Audience Segmentation & Personalisation

User-level profiling, giving recommendations to boost engagement

Timeframe: Medium - Long (3-6 month)

Impact: High – Maximise interest in movies and ROI.

Use Cases

3.2 movies_master.xlsx

- In order to illustrate an initial use case, we use the ‘movies_master.xlsx’, created from the operationalise.py.
- After populating this database for many months and including all historical data, we will get cleaned database to be able to be used in regression.
- In this analysis I have run the operationalise.py after gathering more movies details, I now have a database with 371 film entries, each with the variables: `Title, Release Year, Runtime (mins), IMDb Rating, IMDb Votes, Box Office ($), Age Rating, Google Interest.`
- Using this enriched database we can create a regression model to illustrate how response variables are affected by covariates.

Use Cases

3.3 Regression

- Using R, make regression model using data to model response variable box office earning using covariates (imdb votes omitted).
- On this scale, the model R^2 value is low but not a red flag in itself. Box office revenue is notoriously hard to predict.
- Provides insights, you see that increasing runtime, imdb rating and making the film a PG rating (base level in model, or even better make it G rated) increases the box office earnings.
- This is intuitively sound, illustrating how database can be used.
- Shows scalability, with a larger database with more variables and more complex regression models will able to provide more insights into which movies gross better.

```
> model2.sat.normal = lm(log(box_office) ~., df2);
summary(model2.sat.normal)

Call:
lm(formula = log(box_office) ~ ., data = df2)

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     13.200881   0.977827  13.500  < 2e-16 ***
runtime          0.022267   0.005443   4.091 5.32e-05 ***
imdb_rating      0.332413   0.137676   2.414  0.01626 *
ratingG          0.976691   1.054338   0.926  0.35489
ratingNC-17     -2.792024   1.791164  -1.559  0.11994
ratingNot Rated -4.883803   0.465470 -10.492  < 2e-16 ***
ratingPG-13     -0.994262   0.305336  -3.256  0.00124 **
ratingR         -2.427976   0.282737  -8.587 2.82e-16 ***
ratingTV-PG     -3.055625   1.796159  -1.701  0.08978 .
ratingUnrated   -5.415107   1.301660  -4.160 3.99e-05 ***
google_interest  0.063609   0.012465   5.103 5.46e-07 ***


Residual standard error: 1.775 on 356 degrees of freedom
Multiple R-squared:  0.4179,  Adjusted R-squared:  0.4015
F-statistic: 25.55 on 10 and 356 DF,  p-value: < 2.2e-16
```

Adjusted R-squared: 0.4015 Shows moderately explains variance.

Note: including imdb rating and google_interest likely cause data leakage.

Analysis

3.4 Insights Repeated

1. Do longer films gross more?

```
> cor.test(runtime, log(box_office))

        Pearson's product-moment correlatior

data:  runtime and log(box_office)
t = 3.48, df = 368, p-value = 0.0005615
alternative hypothesis: true correlation is
95 percent confidence interval:
 0.07796092 0.27543728
sample estimates:
     cor
0.178496
```

2. Correlation between IMDb votes and ratings?

```
> cor.test(imdb_votes, imdb_rating)

        Pearson's product-moment correlation

data:  imdb_votes and imdb_rating
t = 13.326, df = 368, p-value < 2.2e-16
alternative hypothesis: true correlation is not
95 percent confidence interval:
 0.4974930 0.6354994
sample estimates:
      cor
0.5705099
```

3. Correlation between google searches and IMDb rating?

```
> cor.test(google_interest, imdb_rating)

        Pearson's product-moment correlation

data:  google_interest and imdb_rating
t = 3.2506, df = 368, p-value = 0.001258
alternative hypothesis: true correlation is not
95 percent confidence interval:
 0.0662407 0.2645146
sample estimates:
      cor
0.1670662
```

4. Correlation between google searches and box office earnings?

```
> cor.test(google_interest, log(box_office))

        Pearson's product-moment correlation

data:  google_interest and log(box_office)
t = 5.9362, df = 368, p-value = 6.747e-09
alternative hypothesis: true correlation is
95 percent confidence interval:
 0.1996804 0.3859380
sample estimates:
     cor
0.295616
```